```
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBB          BBB   AAA       AAA   SSS           RRR         RRR       TTT        LLL
BBBBBBBBBBBB       AAA       AAA     SSSSSSSSS   RRRRRRRRRRR           TTT        LLL
BBBBBBBBBBBB       AAA       AAA     SSSSSSSSS   RRRRRRRRRRR           TTT        LLL
BBBBBBBBBBBB       AAA       AAA     SSSSSSSSS   RRRRRRRRRRR           TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR             TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR             TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR             TTT        LLL
BBB          BBB   AAA       AAA           SSS  RRR      RRR           TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR             TTT        LLL
BBB          BBB   AAA       AAA           SSS  RRR      RRR           TTT        LLL
BBB          BBB   AAA       AAA           SSS  RRR      RRR           TTT        LLL
BBB          BBB   AAA       AAA           SSS  RRR      RRR           TTT        LLL
BBBBBBBBBBBB       AAA       AAA   SSSSSSSSSSS   RRR         RRR       TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA       AAA   SSSSSSSSSSS   RRR         RRR       TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA       AAA   SSSSSSSSSSS   RRR         RRR       TTT        LLLLLLLLLLLLLLL
```

```
BBBBBBBB      AAAAAA      SSSSSSSS  PPPPPPPP      000000    WW      WW  RRRRRRRR    DDDDDDDD
BBBBBBBB      AAAAAA      SSSSSSSS  PPPPPPPP      000000    WW      WW  RRRRRRRR    DDDDDDDD
BB      BB   AA      AA  SS        PP      PP  00      00  WW      WW  RR      RR  DD      DD
BB      BB   AA      AA  SS        PP      PP  00      00  WW      WW  RR      RR  DD      DD
BB      BB   AA      AA  SS        PP      PP  00      00  WW      WW  RR      RR  DD      DD
BB      BB   AA      AA  SS        PP      PP  00      00  WW      WW  RR      RR  DD      DD
BBBBBBBB     AA      AA    SSSSSS  PPPPPPPP    00      00  WW      WW  RRRRRRRR    DD      DD
BBBBBBBB     AA      AA    SSSSSS  PPPPPPPP    00      00  WW      WW  RRRRRRRR    DD      DD
BB      BB   AAAAAAAAAA          SS  PP        00      00  WW  WW  WW  RR  RR      DD      DD
BB      BB   AAAAAAAAAA          SS  PP        00      00  WW  WW  WW  RR  RR      DD      DD
BB      BB   AA      AA          SS  PP        00      00  WWWW  WWWW  RR    RR    DD      DD      ....
BB      BB   AA      AA          SS  PP        00      00  WWWW  WWWW  RR    RR    DD      DD      ....
BBBBBBBB     AA      AA  SSSSSSSS  PP          000000    WW      WW  RR      RR  DDDDDDDD        ....
BBBBBBBB     AA      AA  SSSSSSSS  PP          000000    WW      WW  RR      RR  DDDDDDDD        ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

Table of contents

```
0000     1          .TITLE  BAS$POWRD                ; BASIC float ** double routine
0000     2          .IDENT  /1-001/                  ; File: BASPOWRD.MAR Edit:PLL1001
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28
0000    29 ;++
0000    30 ; FACILITY: Basic Support Library
0000    31 ;
0000    32 ; ABSTRACT:
0000    33 ;
0000    34 ;       This module contains entry points to support exponentiation
0000    35 ;       (** or ^) in BASIC-PLUS-2 for FLOAT ** DOUBLE.
0000    36 ;
0000    37 ; ENVIRONMENT: User Mode, AST Reentrant
0000    38 ;
0000    39 ;--
0000    40 ; AUTHOR: P. Levesque   , CREATION DATE: 5-Oct-81
0000    41 ;
0000    42 ; MODIFIED BY:
0000    43 ;
0000    44 ; 1-001 - Original
```

```
                        .SBTTL  DECLARATIONS

                ;
                ; INCLUDE FILES:
                ;

                ;
                ; EXTERNAL DECLARATIONS:
                ;
0000            .DSABL  GBL                     ; Prevent undeclared
0000                                            ; symbols from being
0000                                            ; automatically global.
0000
0000            .EXTRN  OTS$POWRD               ; OTS$ float ** double exponentiation
0000            .EXTRN  OTS$POWRJ               ; OTS$ float ** int exponentation
0000            .EXTRN  BAS$K_DIVBY_ZER         ; Divide by Zero
0000            .EXTRN  BAS$K_ILLARGLOG         ; Illegal argument in LOG
0000            .EXTRN  BAS$$STOP               ; Error reporting routine
0000
                ;
                ; MACROS:
                ;

                ;
                ; EQUATED SYMBOLS:
                ;

                ;
                ; OWN STORAGE:
                ;

                ;
                ; PSECT DECLARATIONS:
                ;
00000000        .PSECT _BAS$CODE PIC, USR, CON, REL, LCL, SHR, -
0000                        EXE, RD, NOWRT, LONG
```

```
                                        .SBTTL  BAS$POWRD  - BASIC float ** double
                                ;++
                                ; FUNCTIONAL DESCRIPTION:
                                ;
                                ;       This routine takes BASE ** EXP, using the following table
                                ;       for unusual cases:
                                ;
                                ;       BASE > 0                        Call OTS$POWRD, normal case.
                                ;       BASE = 0, EXP > 0               Return 0.0.
                                ;       BASE = 0, EXP = 0               Return 1.0.
                                ;       BASE = 0, EXP < 0               Error: divide by zero
                                ;       BASE < 0, EXP even integer      Call OTS$POWRJ with -BASE
                                ;       BASE < 0, EXP odd integer       Call OTS$POWRJ with -BASE, negate result
                                ;       BASE < 0, EXP not integer       Error: illegal argument in LOG.
                                ;
                                ; CALLING SEQUENCE:
                                ;
                                ;       CALL result.wd.v = BAS$POWRD (base.rf.v, exponent.rd.v)
                                ;
                                ; INPUT PARAMETERS:
                                ;
                        00000004        base = 4
                        0000000C        exponent = 12
                                ;
                                ; IMPLICIT INPUTS:
                                ;
                                ;       NONE
                                ;
                                ; OUTPUT PARAMETERS:
                                ;
                                ;       NONE
                                ;
                                ; IMPLICIT OUTPUTS:
                                ;
                                ;       NONE
                                ;
                                ; FUNCTION VALUE:
                                ; COMPLETION CODES:
                                ;
                                ;       double result of exponentiation
                                ;
                                ; SIDE EFFECTS:
                                ;
                                ;       Will signal Divide By Zero or Illegal argument in LOG if its
                                ;       arguments are bad, and OTS$POWRD and OTS$POWRJ may also signal.
                                ;
                                ;--
                                
                        0000'   BAS$POWRD::     .MASK OTS$POWRD         ; Entry point
                                                                        ; Since this routine uses no
                                                                        ; registers and usually transfers
                                                                        ; control to OTS$POWRD, we copy
                                                                        ; its register save mask and then
                                                                        ; JMP past its save mask and only
                                                                        ; save the registers once
04 AC   53                      TSTF    base(AP)                        ; Test base relationship to zero
   06   15                      BLEQ    1$                              ; If base leq 0, do case analysis
```

```
                00000002'GF   17  0007   140          JMP     G^OTS$POWRD+2           ; Transfer control to the OTS$
                                  000D   141                                          ; routine to do exponentiation
                                  000D   142 ;+
                                  000D   143 ; Come here if the base is less than or equal to zero.  We must filter
                                  000D   144 ; several special cases, as described above.
                                  000D   145 ;-
                          2E  13  000D   146 1$:      BEQL    4$                      ; Branch if base = 0
    50   50   08   00   0C AC  74  000F   147          EMODD   exponent(AP), #0, #1, R0, R0
                          1A  12  0016   148          BNEQ    3$                      ; Branch if exponent is not integer
                                  0018   149 ;+
                                  0018   150 ; The base is less than zero and the exponent is an integer.
                                  0018   151 ; BASIC defines this as working the same way as if an integer was
                                  0018   152 ; in the expression (making a double variable which happens to
                                  0018   153 ; contain an integer value equivalent to an integer variable).
                                  0018   154 ;-
                     50   0C AC  6A  0018   155          CVTDL   exponent(AP), R0        ; Convert exponent to integer
                             50   DD  001C   156          PUSHL   R0                      ; Save for even/odd test
                             50   DD  001E   157          PUSHL   R0                      ; Stack as parameter to OTS$POWRJ
                        7E   04 AC  52  0020   158          MNEGF   base(AP), -(SP)         ; Stack -base also
                00000000'GF   03  FB  0024   159          CALLS   #3, G^OTS$POWRJ         ; Call integer power routines
                           03 8E  E9  002B   160          BLBC    (SP)+,2$                ; Branch if exponent even
                        50   50  72  002E   161          MNEGD   R0, R0                  ; Exponent odd, negate the result
                             04  0031   162 2$:      RET                             ; and return with it.
                                  0032   163 ;+
                                  0032   164 ; Come here if the base is less than zero but the exponent is not
                                  0032   165 ; an integer.  BASIC defines this as an error.
                                  0032   166 ;-
                        7E   00'8F  9A  0032   167 3$:      MOVZBL  #BAS$K_ILLARGLOG, -(SP) ; Illegal Argument in LOG
                00000000'GF   01  FB  0036   168          CALLS   #1, G^BAS$$STOP         ; Never return.
                                  003D   169 ;+
                                  003D   170 ; Come here if the base is equal to zero.  The value we return depends
                                  003D   171 ; upon the sign of the exponent.
                                  003D   172 ;-
                     0C AC   73   003D   173 4$:      TSTD    exponent(AP)            ; Test the exponent against zero
                        09   19   0040   174          BLSS    6$                      ; Branch if exponent lss 0
                        03   13   0042   175          BEQL    5$                      ; Branch if exponent is 0
                                  0044   176 ;+
                                  0044   177 ; Come here if the base is zero and the exponent is greater than zero.
                                  0044   178 ; BASIC defines this as 0.0.
                                  0044   179 ;-
                        50   7C   0044   180          CLRD    R0                      ; R0, R1 = 0.0
                             04   0046   181          RET                             ; Return to caller
                                  0047   182 ;+
                                  0047   183 ; Come here if the base is zero and the exponent is zero.  BASIC defines
                                  0047   184 ; this as 1.0.
                                  0047   185 ;-
                     50   08   70   0047   186 5$:      MOVD    #1, R0                  ; R0, R1 = 1.0
                             04   004A   187          RET                             ; Return to caller.
                                  004B   188 ;+
                                  004B   189 ; Come here if the base is zero and the exponent is less than zero.
                                  004B   190 ; BASIC defines this as an error.
                                  004B   191 ;-
                   7E   00'8F   9A   004B   192 6$:      MOVZBL  #BAS$K_DIVBY_ZER, -(SP) ; Divide by zero
            00000000'GF   01   FB   004F   193          CALLS   #1, G^BAS$$STOP         ; Report error, never return.
                                  0056   194 ;
                                  0056   195          .END
```

```
BAS$$STOP           ********    X   00
BAS$K_DIVBY_ZER     ********    X   00
BAS$K_ILLARGLOG     ********    X   00
BAS$POWRD           00000000 RG     01
BASE              = 00000004
EXPONENT          = 0000000C
OTS$POWRD           ********    X   00
OTS$POWRJ           ********    X   00
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
_BAS$CODE	00000056 (86.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.08	00:00:00.56
Command processing	116	00:00:00.45	00:00:02.12
Pass 1	71	00:00:00.50	00:00:01.25
Symbol table sort	0	00:00:00.00	00:00:00.00
Pass 2	47	00:00:00.38	00:00:00.85
Symbol table output	2	00:00:00.02	00:00:00.02
Psect synopsis output	3	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	272	00:00:01.46	00:00:04.82

The working set limit was 900 pages.
2216 bytes (5 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 8 non-local and 6 local symbols.
195 source lines were read in Pass 1, producing 8 object records in Pass 2.
0 pages of virtual memory were used to define 0 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLIB]STARLET.MLB;2	0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

```
MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS$:BASPOWRD/OBJ=OBJ$:BASPOWRD MSRC$:BASPOWRD/UPDATE=(ENH$:BASPOWRD)
```

BASOPEN
LIS

BASPOWJJ
LIS

BASPOS
LIS

BASPOWDJ
LIS

BASOPENDE
LIS

BASPOWGG
LIS

BASPOWHH
LIS

BASPOWRJ
LIS

BASPOWTI
LIS

BASPURIOB
LIS

BASPOWDO
LIS

BASOPENZE
LIS

BASPOWOR
LIS

BASPOWG
LIS

BASPOWRD
LIS

BASPOWH
LIS

BASPOWRR
LIS